

用遗传算法求解 CTSP

喻 镒, 凌 捷, 谢晓峰

(广东工业大学 计算机科学与工程一系, 广东 广州 510090)

摘要: 给出了对于 CTSP 的基于遗传算法的一种最优求解.

关键词: 遗传算法; CTSP 问题; 图论

中图分类号: TP311

文献标识码: A

文章编号: 1007-7162(2000)03-0052-04

生物的进化是一个奇妙的优化过程, 它通过选择淘汰, 突然变异, 基因遗传等规律产生适应环境变化的优良品种. 遗传算法是根据生物进化思想而启发得出的一种全局优化算法. 遗传算法在本质上是一种不依赖具体问题的直接搜索方法, 在模式识别、神经网络、图像处理、机器学习、工业优化控制、生物科学、社会科学等方面都得到了成功的应用.

CTSP(China Travelling Salesman Problem)问题是指从某一城市出发, 求经过中国 31 个城市各一次, 并返回出发点, 所形成的一条路程最短的回路^[1]. CTSP 是一个 NP 完全问题, 有许多近似方法求解它, 如最邻近法、重绕生成树法、分支定界法、最小权匹配算法等. 近几年来, 又有许多国内外学者提出了解决该问题的新的近似算法. 如 Hopfield 网络方法^[2], 几何分块法^[1]、均值比较法^[3]等, Hopfield 网络方法较复杂, 几何分块法则要求问题具有几何特性. 本文利用遗传算法的特性, 提出了一种解决 CTSP 的新算法. 该算法从一个可能解出发进行局部搜索, 经过反复的交换, 变异, 直到满足终止条件, 达到最优. 与其它算法比较, 该算法覆盖可能解的面比较大, 利于找出全局最优, 而且对可能解的反复变异和交换使得该算法具有极强的容错能力.

1 经典遗传算法

遗传算法根据适者生存, 优胜劣汰等自然进化原则, 从候选解中一代一代地优选适应性函数(Fitness Function)的解, 重组后构成新的参数组合, 而逐渐取得最优解^[4]. 适应性函数与传统优化算法中的目标函数和成本函数相对应, 父辈个体(Individual)对子代的贡献与适应性函数成正比. 它通常使用 3 种操作符(Operator): 选择(Selection), 交换(Crossover), 变异(Mutation). 相关概念如下:

(1) 适应性函数

适应性函数应能反映优化目标和约束情况, 并且性能函数值越小, 则适应函数越大.

(2) 初始化候选解

选择一个集合 $\{b_i | i = 1, 2, \dots, n\}$, 这个集合也就是问题假设解的初始群体. 通常以随机方

收稿日期: 1999-11-18

基金项目: 广东省自然科学基金资助项目(993485).

作者简介: 喻镒(1971-), 男, 在读研究生; 主要研究方向: 算法理论及计算机网络安全.

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved. <http://www.>

法产生该集合. 问题的最优解将通过这些初始假设解进化而求出.

(3)选择

在选择时, 以适应度为选择原则, 适应度较高的个体, 繁殖下一代的数目较多; 适应度较小的个体, 繁殖下一代的数目较少, 甚至被淘汰.

(4)交换

对于选中用于繁殖下一代的个体, 随机地选择两个个体的相同位置, 按交叉概率 P_c 实行交换.

(5)全局最优收敛

当最优个体的适应程度达到阈值, 或者最优个体的适应度不再上升时, 则算法结束. 否则, 用经过选择, 交换, 变异所得到的新一代群体取代上一代群体, 并返回到选择操作处继续执行.

2 用遗传算法求解 CTSP

用整数集 $S = \{0, 1, \dots, 30\}$ 表示中国 31 个直辖市和省会城市, 即(北京, 上海, 广州...), 记 X 为 S 上所有排列($v_0, v_1, \dots, v_{30}$)的集合, 每一个排列表示一条从城市 v_0 开始, 依次经过城市 $v_1, v_2, \dots, v_{30}$, 最后返回城市 v_0 的遍历回路, X 也即为候选解的所有编码. 定义性能函数为:

$$P(X) = \sum d(v_i, v_{i+1}) + d(v_{30}, v_0),$$

其中 v_i 为 X 中的元素, $d(v_i, v_{i+1})$ 表示城市 v_i 到城市 v_{i+1} 的距离. 定义适应性函数为:

$$\text{Adapt}(X) = \text{comp} - P(X),$$

式中 comp 为父体(即刚进行过变换的候选解)的性能函数值. 若 $\text{Adapt}(X) \geq 0$, 则此候选解可作为父体再次进行迭代; 否则, 淘汰此解.

为了引入遗传算法, 必须定义变异和交换操作. 这里利用城市编号进行编码, 为了避免交换操作后的乱码现象, 在此采用单性繁殖^[5], 即只对单个父体进行操作. 所采用的操作是每次只有一个候选解, 在编码内截取一段代码, 对这段代码进行全排列, 选择所有排列中最优的一组, 根据适应度函数决定是否放回原编码中以形成新的编码.

确定所截取的一段代码的区间的原则如下:

(1)领域中心 α 的确定:

定义 1 $D(\alpha, \beta)$ 为 α 的 β 领域, 其中 β 是领域半径. β 越大, $D(\alpha, \beta)$ 中所包含元素就越多, 同时 $D(\alpha, \beta)$ 是对称的, 即 $X_1 \in D(X_2, \beta)$ 当且仅当 $X_2 \in D(X_1, \beta)$ ^[6].

本算法中 α 指向父体所表达的回路中最大的一条边的起点, 若父体中最大边与父体的父体最大边相同, 则 α 指向第二大边, 依次类推. 这样做的目的是利于从局部极小点趋向全局最小点.

(2)领域半径 β 的确定:

定义 2 对比度 $\delta = (\text{comp} - \text{dist})/e$,

式中: 对比度 δ 是反映了一个可能解优劣程度的相对指标, δ 值越小, 说明越逼近局部极小值, 解已达到较优; $e = (\max - \min)$, $\min$ 为 31 城市间所有 465 条边中最小的 31 条边之和, $\max$ 为 31 城市间所有 465 条边中最大的 31 条边之和; dist 为本次变异中得到的一组最优排列的性能函数值.

根据遗传算法, 适应度较高的个体应繁殖较多的后代, 从而从局部极小转向全局最小. 为此, 本算法构造了一个从 δ 到 β 的映射, 如图 1.

遗传算法的结束条件是:直到父体中所有点的领域均作过变异且子代均未达到适应度函数的规定.

3 算法实现

根据上述思路,作者用 C 语言实现了对 CTSP 的求解,主要程序代码如下:

```
void main()  
{ Initial();  
  β= Reflect();  
  dist= Permu(α,β);  
  i= 0;  
  do  
  { flag= 0;  
    while(i< 31)  
    { α= Find(i);  
      if( α= mark)  
        i ++ ;  
      else  
        break;  
    }  
    mark= α;  
    dist= permu(α,β);  
    if( dist< comp)  
    { i= 0;  
      β= Reflect();  
      comp= dist;  
      以子代更新父体;  
      flag= 1;  
    }  
  }while( flag || i< 30);  
  输出结果;  
}
```

其中, Initial 为初始化函数,它根据 31 个节点(城市)的 465 条边,按从小到大的顺序选择,直到选中的 31 条边能构成回路. Permu 函数对选定的变异区间进行全排列,输出本次变换中最优的一组解的性能函数值. Find 函数找出排列所形成的回路的所有边中的第 i 大边的起点. Reflect 函数实现从 δ 到 β 的映射.

利用本算法对中国 TSP 问题求解,其路径为:

(北京—呼和浩特—太原—石家庄—郑州—西安—银川—兰州—西宁—乌鲁木齐—拉萨—成都—昆明—贵阳—南宁—海口—广州—长沙—武汉—南昌—福州—台北—杭州—上海—南

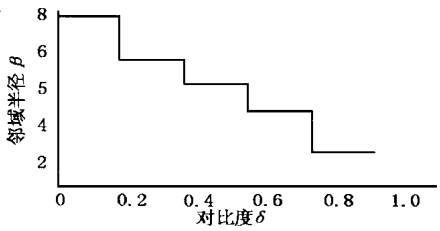


图 1 邻域半径大小映射图

京—合肥—济南—天津—沈阳—长春—哈尔滨)

获得结果的总路程为 15 404 km, 这也是 CTSP 问题的最短路径^[1].

4 算法分析

本算法作为一种近似计算方法, 与其他求解 TSP 问题的算法比较, 具有如下特性,

(1) 编码简单, 易于实现. 本算法避免了经典遗传算法的随机性, 它无需根据实际情况来选取交换概率和变异概率.

(2) 由本算法构造的初始解出发, 经过所定义交换操作所形成的解空间一定包含了最优解, 这样只要从 δ 到 β 的映射图构造恰当, 就一定能求出最优解.

(3) 容易快速找出局部最优解. 由于每一次交换操作都在父体的最不优良的因素周围进行, 即在父体回路中的最大一条边的节点周围进行全排列, 而且父体越优良, 这种排列操作的幅度越大.

(4) 具有局部穷举特性, 但时间复杂度较穷举法大为降低. 如在本例中, 若以计算一条回路长度为基本操作, 则本算法在最不利情况下需要计算 $31^* 8!$ 次, 而穷举法则需 $31!$ 次.

5 结束语

TSP 问题是一个 NP 完全问题, 目前国内外还不能找出一个多项式算法解决此类问题, 但解决此类问题有较大的现实意义. 例如: 在网络通信中求解路由问题时, 可用节点间的路径长度代表网络节点间的通信代价, 而求解一条代价最小路由回路即可转化为 TSP 问题的求解. 本算法实现了对 CTSP 的最优求解, 对其他具体的 TSP 问题也同样适用, 具有较强的实用性. 针对不同的情况, 只要更改从 δ 到 β 的映射图即可.

参考文献:

- [1] 陈沐天, 蔡和熙. 货郎担问题的几何分块算法及 China TSP 问题的最终解决[J]. 计算机工程与科学, 1998, (1): 22-27.
- [2] 孙守坚, 郑钧里. Hopfield 网络求解 TSP 的一种改进算法和理论证明[J]. 电子学报, 1995, (1): 73-78.
- [3] 凌捷, 张吉文, 陈文戈. 求解 TSP 的一种新算法[J]. 计算机应用研究, 1999, (4): 52-54.
- [4] 余永权. 神经网络模糊逻辑控制[M]. 北京: 电子工业出版社, 1999. 232-238.
- [5] 段国林, 查建中, 林建平, 等. 遗传算法在钟表机芯设计中的应用[J]. 软件学报, 1998, (7): 515-519.
- [6] 于志伟, 陶波, 汪元美. 一种竞争算法及其在组合优化中的应用[J]. 软件学报, 1998, (10): 794-796.

Solving CTSP with Genetic Algorithm

YU Di, LING Jie, XIE Xiao-feng

(Dept. of Computer Science and Engineering, GDUT, Guangzhou 510090, China)

Abstract: An optimizing solution to the CTSP was proposed based on genetic algorithm in this paper.

Key words: genetic algorithm; CTSP; graph theory